Hochschule Bochum
Bochum University
of Applied Sciences

Fachbereich Wirtschaft
Prof. Dr. Christian Bockermann

Data Science 1

Wintersemester 2022 / 2023

Hausarbeit

Die Prüfungsleistung zum Modul *Data Science 1* findet als Hausarbeit statt. Die Aufgabenstellung zur Hausarbeit finden Sie in diesem Dokument.

Für die Bearbeitung der Aufgabenstellung und die Erstellung Ihrer Hausarbeit steht wieder der Jupyter-Notebook Server zu Verfügung. Die Abgabe der Hausarbeit erfolgt dann als PDF-Export Ihres Jupyter-Notebooks. Das PDF Ihres Notebooks schicken Sie bis zum 26.2.2023 per Mail an: `christian.bockermann@hs-bochum.de`
Andere Formen der Abgabe sind nicht vorgehen.

Als Materialien können Sie sämtliche Unterlagen aus der Vorlesung und den Übungen mit benutzen, im Internet recherchieren oder weitere Bücher/Kurse mit verwenden. Geben Sie bitte bei Verwendung von umfangreicherem Programm-Code aus dem Netz (mehr als 3-4 Zeilen) die Quelle kurz mit an.

Aufgabe 1 (Python Basics)

Klemmbausteine der Firma LEGO® erfreuen seit Jahrzehnten die Kinderherzen und werden auch in der Bildung für die spielerische Lernerfahrung eingesetzt. Die Firma LEGO® bringt jedes Jahr neue Sets heraus, die zu unterschiedlichen Themenbereichen gehören (z.B. City oder Technic), für unterschiedliche Altersgruppen konzipiert sind und verschiedene Anzahlen an Steinen enthalten.
Dafür liegt eine Liste der Sets seit 1980 vor, über 20.000 verschiedene Sets enthält. Die Elemente der Liste haben das folgende Format:

```
(nummer, name, thema, jahr, teile)
```

Die **nummer** ist die Produktnummer des Sets, **name** eine textuelle Beschreibung und **thema** die Themenwelt, der das Set zugeordnet ist. Das Attribut **jahr** enthält das Erscheinungsjahr des Sets. Zusätzlich ist **teile** die Anzahl der im Set enthaltenen Steine gespeichert.
Auf dem Jupyter Notebook Server steht Ihnen das Modul **bricks** zur Verfügung, das eine Funktion **set_list()** enthält, die die Liste der Songs liefert:

```
import bricks

sets = bricks.set_list()

sets[712]  #  ('30366-1', 'Police Car', 'City', 2020, 37)
```

Wie in dem Python Code zu sehen ist, hat beispielsweise der Song am Index 98 der Liste die folgenden Werte:

```
('30366-1', 'Police Car', 'City', 2020, 37)
```

Das heisst, das Set *30366-1* ist ein Polizeiauto Modell aus dem Thema *Lego City*. Es ist 2020 erschienen und enthält 37 Teile.

Für eine derartige Liste sollen Sie die folgenden Aufgaben lösen:

1. Bestimmen Sie die *verschiedenen* Themen aus der Liste der LEGO® Sets!
   Schreiben Sie dazu eine Funktion `themen(xs)`, die für die Liste xs von Sets die Menge der Themen zurückgibt, die in der Liste enthalten sind.

   Das Ergebnis der Funktion soll also ein Liste oder Menge (**set**) sein, die jedes Thema nur einmal enthält.

2. Schreiben Sie eine Funktion `setsZuThema(xs, thema)`, die für die Liste **xs** aller Sets und das Thema **thema**, die Liste der Sets zurückliefert, die zu diesem Thema gehören.

3. Schreiben Sie eine Funktion `anzahlNachThema(xs)`, die die Liste **xs** der Sets nach den Themen gruppiert und für jedes Thema die Anzahl der Sets berechnet.

   Die Liste soll also folgendes Format haben:

   ```
   [('Gear', 2592), ('Sculptures', 7), ('Pirates', 90),...]
   ```

4. Schreiben Sie eine Funktion `avgTeile(xs, thema)`, die für die Liste `xs` und das Thema `thema` die durchschnittliche Anzahl der Teile für die Sets berechnet. Dabei sollen Sets mit 0 Teilen nicht mit in die Berechnung einfließen!!

5. Schreiben Sie eine Funktion `meisteSets(xs)` die jeweils die Themen mit den meisten Sets zurückliefert.

   (Sofern es mehrere Themen mit der gleichen maximalen Anzahl von Sets gibt, sollen alle diese Themen zurückgegeben werden. Ihre Funktion muss also eine Liste zurückgeben.)

Aufgabe 2 (Pandas und Statistiken)

Die Datei `Kurse/DataScience1/data/bricksets.csv` enthält eine etwas umfassendere Datenbasis von LEGO® Sets als in der ersten Aufgabe. In der nachfolgenden Tabelle ist ein kleiner Auszug abgebildet, der unter anderem auch die Altersgruppe enthält.

Nummer	Name	Jahr	Thema	Altersgruppe	AnzahlTeile
6870-1	Space Probe Launcher	1981	Space	6+	60
8848-1	Power Truck Unimog	1981	Technic	10+	398
6611-1	Fire Chief's Car	1981	Town	6+	20
3701-1	Fisherman Cornelius ...	1982	Fabuland	6+	3
3703-1	Peter Pig the Cook	1982	Fabuland	4+	3
3704-1	Marjorie Mouse	1982	Fabuland	4+	1
3707-1	Clover Cow	1982	Fabuland	4+	3
3708-1	Rufus Rabbit	1982	Fabuland	8+	4
3709-1	Henry Horse, Carpent...	1983	Fabuland	4+	3

Die Bedeutung der Spalten ist eigentlich selbsterklärend. Anhand dieses Datensatzes läßt sich allerdings die Entwicklung der Sets der Firma LEGO® etwas verfolgen. Das heisst wir sind in dieser Aufgabe auf der Suche nach Fragen wie z.B.:

- Wie hat sich die Anzahl der Sets über die Jahre entwickelt?
- Wann sind neue Themen-Gebiete dazu gekommen?
- Gibt es bestimmte Altergruppen, für die über die Jahre mehr Sets produziert wurden?

Hintergrund dieser Aufgabe ist es, dass Sie sich mit einem unbekannten Datensatz vertraut machen und mit Hilfe von Pandas untersuchen, welche Informationen aus den Daten herausgesucht werden können.

Die Aufgaben:

1. Zunächst sollen ein paar generelle Informationen berechnet werden:
   - Wieviele Sets gibt es in dem Datensatz? Über welchen Zeitraum sind überhaupt Daten verfügbar?
   - Gibt es fehlende Werte? Gibt es Duplikate in den Sets?
   - Wieviele und welche Themen wurden von der Firma LEGO® über die Jahre veröffentlicht?
   - Welche Altersgruppen gibt es?
2. Erstellen Sie ein Diagramm, das die Anzahl der veröffentlichten Sets pro Jahr darstellt.

3. Wie hat sich die durchschnittliche Anzahl der Teile für Sets der Themen *Technic* und *X* entwickelt? Plotten Sie diese Durchschnittwerte für die beiden Themen.

   (Wählen Sie für *X* eines der übrigen Themen. Sie können auch verschiedene ausprobieren.)

   Hinweis: Haben alle Sets immer auch Steine? Sets ohne Steine verzerren den Durchschnitt. Nehmen Sie diese für die Durchschnittsberechnung heraus.

4. Im Folgenden soll die Entwicklung der Themen über die Jahre betrachtet werden. Berechnen Sie die Anzahl der Sets für jedes Jahr und jedes Thema. Dazu ist es hilfreich, aus der Spalte *Thema* weitere Spalten der Art

Nummer	...	**Thema_Space**	**Thema_Fabuland**	**Thema_Technic**	...
6870-1	..	1	0	0	...
8848-1	..	0	0	1	...
6611-1	..	0	0	0	...
3701-1	..	0	1	0	...
3703-1	..	0	1	0	...

   zu berechnen.

   - Ab wann (Jahr) wurde das Thema *Ninjago* populär?
   - Wieviele Sets pro Jahr im Durchschnitt für das Thema *Technic*?
   - In welchem Jahr gab es für *Technic* die meisten Sets?
   - Welche Themen haben die Sets mit den meisten Steine im Durchschnitt?
   - Wie hat sich die durchschnittliche Zahl der Steine über die Jahre entwickelt? Erstellen Sie dazu einen Plot! (Sets ohne Steine sollen dazu wieder nicht betrachtet werden.)

   Tipp: Schauen Sie sich für die Berechnung der Themen-Spalten die Funktion `get_dummies` aus dem Pandas Modul an. Wenn Sie das Pandas Modul als `pd` importiert haben, können Sie mit dem Befehl

   ```
   help(pd.get_dummies)
   ```

   die interaktive Hilfe zu dieser Funktion im Notebook aufrufen.

5. Wählen Sie 6 beliebige Themen aus und berechnen Sie für jedes dieser Themen den prozentualen Anteil von erschienen Sets im Bezug auf die Gesamtzahl pro Jahr.

6. Berechnen Sie aus der Spalte *Altersgruppe* eine neue Spalte *abAlterX*, die das jeweilige Mindestalter der Altersgruppe enthält.

Berechnen Sie nun für jedes Mindestalter die durchschnittliche Anzahl der Steine in den Sets, sowie die jeweilige Maximalanzahl.

Wie verläuft das Verhältnis von Mindestalter zu maximaler Anzahl von Steinen? Erstellen Sie dazu einen Plot.

Aufgabe 3 (Modell-Training)

Um den Verkauf von Spielwaren anzukurbeln, setzen die großen Plattformen und Hersteller auf online Werbung. Jede eingeblendete Werbung kann einen Internet-Benutzer zum Kauf verleiten - erzeugt aber auch Kosten. Daher ist es natürlich wichtig zu entscheiden, wem man Werbung einblendet und bei wem man sich dies ggf. sparen kann - für eine kosteneffektive Werbestrategie.
Im Folgenden betrachten wir einen fiktiven Datensatz zur Online-Werbung. Der Datensatz enthält Merkmale von Internet-Users (z.B. Alter, Internet-Nutzung, etc.) und in der Spalte `Thema` das Thema des Banners, das dem Benutzer präsentiert wurde. Die Spalte `click` enthält eine 1, wenn der Benutzer auf das Banner geklickt hat (=gekauft hat) und eine 0 sonst.
Den Datensatz finden Sie in der Datei

`Kurse/DataScience1/data/lego-adverts.csv`

Die Aufgabe ist nun, ein Vorhersagemodell zu erzeugen, dass vorhersagt, ob ein Benutzer mit den gegebenen Eigenschaften durch das Werbebanner zum Kauf verleitet wird oder nicht. Dafür soll im folgenden ein Entscheidungsbaum-Modell trainiert werden.

1. Laden Sie die Daten in einen DataFrame und geben Sie die Anzahl der Datensätze, sowie die Anzahl der verschiedenen Themen, für die Werbung eingeblendet wurde.

   Betrachten Sie die Spalten und Datentypen der Spalten und überlegen Sie sich, welche Spalten Sie für ein Vorhersagemodell verwenden wollen. Da der Entscheidungsbaum erstmal nur mit numerischen Daten umgehen kann, kann die Spalte `Thema` nicht direkt benutzt werden.

   Benutzen Sie wieder die Funktion `pd.get_dummies(..)` um diese Spalte in mehrere 0/1 Spalten zu überführen.

2. Wie hoch ist der Fehler eines Modells, das immer nur *Benutzer kauft* vorhersagt?

3. Trainieren Sie ein Entscheidungsbaum-Modell auf dem vollständigen Datensatz. Welchen Trainingsfehler erreicht ihr Modell?

4. Teilen Sie die Daten in Trainings- und Test-Daten auf und verwenden Sie dabei 80% der Daten zum Training und den restlichen Teil für das Testen.

5. Bestimmen Sie den Parameter `max_depth`, der auf den Daten für ein Entscheidungsbaum-Modell den besten Generalisierungsfehler liefert. Testen Sie für die Bestimmung des Parameters `max_depth` die Werte im Bereich von 2 bis 10.

   Erzeugen Sie dazu einen DataFrame, der den Parameter `max_depth`, den Trainings- und den Test-Fehler enthält.

6. Erstellen Sie einen Plot mit dem Parameter `max_depth` auf der x-Achse, der den Trainings- und Test-Fehler für die verschiedenen Werte von `max_depth` zeigt. Für welches `max_depth` bekommen Sie das beste Modell?

7. Erweitern Sie Ihren Code so, dass für jeden Parameterwert `max_depth` mehrfache Train/Test Splits durchgeführt werden. Das bedeutet, jeder Parameterwert soll mit 5 verschiedenen Train/Test Splits evaluiert werden. Die Ergebnisse sollen dann wieder in einem DataFrame zusammengeführt werden.

   Berechnen Sie aus dem resultierenden DataFrame dann die durchschnittlichen Trainings- und Test-Fehler für jeden Parameterwert und erstellen Sie einen Plot dazu.

   Tipp: Dadurch verlängert sich natürlich auch die Laufzeit des Programmes auf das 5-fache. Für die Entwicklung ist es daher hilfreich, sich vielleicht auf z.B. nur 4 Parameterwerte (2 bis 5) und jeweils 2 Train/Tests Splits zu beschränken. Diese Werte können Sie ja dann für den finalen Lauf anpassen.